PAPER

Stochastic Dykstra Algorithms for Distance Metric Learning with Covariance Descriptors

Tomoki MATSUZAWA[†], **Eisuke ITO**[†], **Raissa RELATOR**[††], **Jun SESE**[††], ***Nonmembers***, ***and*** **Tsuyoshi KATO**[†a)], ***Member***

SUMMARY In recent years, covariance descriptors have received considerable attention as a strong representation of a set of points. In this research, we propose a new metric learning algorithm for covariance descriptors based on the Dykstra algorithm, in which the current solution is projected onto a half-space at each iteration, and which runs in $O(n^3)$ time. We empirically demonstrate that randomizing the order of half-spaces in the proposed Dykstra-based algorithm significantly accelerates convergence to the optimal solution. Furthermore, we show that the proposed approach yields promising experimental results for pattern recognition tasks.

key words: *covariance descriptor, metric learning, convex optimization, stochastic optimization, Dykstra algorithm*

1. Introduction

Learning through example objects characterized by a set of several points instead of one point in a feature space is an important task in computer vision and pattern recognition. In recent years, covariance descriptors [1]–[4] have received considerable attention as a strong representation of a set of points. The performance of categorizing covariance descriptors depends on the metric that is used to measure the distances between them. To compare covariance descriptors, various distance measures, such as affine invariant Riemannian metric [5], Stein metric [6], J-divergence, Frobenius distance [1], and log-Frobenius distance [7], have been discussed in existing literature. A few of these are designed from their geometrical properties; however, others are not. Several of these distance measures are expressed in the form

$$D_{\mathbf{\Phi}}(X_1, X_2) := \|\mathbf{\Phi}(X_1) - \mathbf{\Phi}(X_2)\|_{\mathrm{F}}^2, \tag{1}$$

where $\mathbf{\Phi}$ is a function that maps a symmetric positive definite matrix to a square matrix of the same size. If $\mathbf{\Phi}(X) := \mathbf{logm}(X)$, where $\mathbf{logm}(X)$ is the principal matrix logarithm of a strictly positive definite matrix, X, the log-Frobenius distance [7] is obtained. Setting $\mathbf{\Phi}(X) := X^p$ yields the power-Frobenius distance [1], while $\mathbf{\Phi}(X) := \mathbf{chol}(X)$, where $\mathbf{chol}(\cdot)$ produces the Cholesky decomposition of X such that $X = \mathbf{chol}(X)\mathbf{chol}(X)^\top$, yields the Cholesky–Frobenius distance [8]. These metrics are predefined before employing machine learning algorithms, and are not adaptive to the data to be analyzed. For categorizing vectorial data, supervised learning for fitting metrics to a task significantly enhances the performance of distance-based classifiers [9]–[11].

In this study, we introduce a parametric distance measure between covariance descriptors, and propose novel metric learning algorithms to determine the parameters of the distance measure function. The learning problem is formulated as the Bregman projection onto the intersections of half-spaces. This problem can be solved using the Dykstra algorithm [12], [13], which selects a half-space in cyclic order and projects the current solution onto the half-space. We developed an efficient technique for projection onto a half-space. Furthermore, we empirically found that selecting the half-space stochastically, rather than in cyclic order, significantly increases the speed of convergence to an optimal solution.

Related work. Jayasumana et al. [1] defined a positive definite kernel among covariance descriptors based on the following two facts: the exponential of a conditional negative definite kernel is positive definite, and the distance function given in the form of (1) is a conditional negative definite kernel. A disadvantage of using such a kernel is that the feature space derived from the kernel is not always provided explicitly. A few approaches reduce dimensionality [14]–[16] by finding an informative subspace or learning a more discriminant lower dimensional space. However, in these methods, information relevant to the discrimination may be discarded when projecting onto a lower dimensional subspace, which may reduce classification performance.

Vamulapalli and Jacobs [3] introduced a supervised metric learning approach for covariance descriptors. They vectorized the matrix logarithms of covariance descriptors to apply existing metric learning methods to the vectorizations of matrices. The dimensionality of the vectorizations is $n(n+1)/2$ when the sizes of the covariance matrices are $n \times n$. Thus, the size of the Mahalanobis matrix is $n(n+1)/2 \times n(n+1)/2$, which is computationally prohibitive when n is large.

The proposed approach is a generalization of the distance measure proposed by Huang et al. [2], which is based on the Log-Euclidean metric, with the loss function being a special case of the proposed formulation. They used the cyclic Dykstra algorithm for learning the Mahalanobis-like

Manuscript received July 29, 2016.
Manuscript revised November 27, 2016.
Manuscript publicized January 13, 2017.

[†]The authors are with the Department of Computer Science, Gunma University, Kiryu-shi, 376–8515 Japan.
[††]The authors are with the AIST Artificial Intelligence Research Center, Tokyo, 135–0064 Japan.
a) E-mail: katotsu@cs.gunma-u.ac.jp

DOI: 10.1587/transinf.2016EDP7320

matrix. Their primary finding is that projection onto a half-space can be obtained analytically. However, their claim does not hold because of incorrectly using the Woodbury matrix inversion formula and deriving an incorrect closed-form solution (In [2], $\boldsymbol{A}_k$ was treated as a one-rank matrix and the Woodbury inversion formula was incorrectly applied.). Therefore, their algorithm does not ensure theoretical convergence to the optimal solution. In this study, the update rule that they used is corrected by presenting a new technique that projects the current solution onto a half-space within $O(n^3)$ computational time.

Yger and Sugiyama [4] devised a different formulation of metric learning. They introduced a congruent transform and measured the distances between the transformations of covariance descriptors. An objective function based on kernel target alignment [17] is employed to determine transformation parameters. Compared to their algorithm, the proposed algorithm can monitor the upper bound of an objective gap, i.e., the difference between the current and minimum objectives. This implies that the resultant solution is ensured to be ϵ-suboptimal if the convergence criterion of the algorithm is set such that the upper bound of the objective gap is less than a very small number, ϵ. As Yger and Sugiyama [4] employed a gradient method for learning the congruent transform, the objective gap could not be determined.

Contributions. The contributions of this study can be summarized as follows:

- For metric learning on a positive semidefinite cone, we developed a new algorithm based on the Dykstra algorithm, in which the current solution is projected onto a half-space at each iteration, and which runs in $O(n^3)$ time (Fig. 1).
- We empirically determined that randomizing the order of half-spaces in the proposed Dykstra-based algorithm significantly accelerates convergence to the optimal solution. In particular, the proposed approach allows for using an *almost hard margin* (weak regularization), whereas the classical approach can attain an optimum within a practical time only when a strong regularization is selected.
- We propose an upper bound for the objective gap, which provides a stopping criterion and ensures optimality of the solution.
- We show that the proposed approach yields promising experimental results for pattern recognition tasks.

All proofs can be found in the supplements of our conference paper [18] and our technical report [19]. The difference between the current version and [18] is the addition of a new experimental result on electroencephalogram (EEG) signal classification.

2. Metric Learning Problem

We introduce the following dissimilarity measure for covariance descriptors $\boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathbb{S}_+^n$:

$$D_{\boldsymbol{\Phi}}(\boldsymbol{X}_1, \boldsymbol{X}_2; \boldsymbol{W}) := \left\langle \boldsymbol{W}, (\boldsymbol{\Phi}(\boldsymbol{X}_1) - \boldsymbol{\Phi}(\boldsymbol{X}_2))(\boldsymbol{\Phi}(\boldsymbol{X}_1) - \boldsymbol{\Phi}(\boldsymbol{X}_2))^\top \right\rangle,$$

where $\boldsymbol{W} \in \mathbb{S}_+^n$ is the parameter of this function. $\mathbb{S}_+^n$ and $\mathbb{S}_{++}^n$ denote the sets of $n \times n$ positive semi-definite matrices and strictly positive semi-definite matrices, respectively, and $\mathbb{R}_+^n$ and $\mathbb{R}_{++}^n$ denote the sets of n-dimensional real vectors with non-negative and strictly positive entries, respectively.

Theorem 2.1: If $\boldsymbol{W}$ is strictly positive definite and $\boldsymbol{\Phi}$ is bijective, then, the dissimilarity measure, $D_{\boldsymbol{\Phi}}(\cdot, \cdot; \boldsymbol{W}) : \mathbb{S}_+^n \times \mathbb{S}_+^n \to \mathbb{R}$, is a distance metric.

To determine the value of the parameter matrix, $\boldsymbol{W}$, we pose a constrained optimization problem based on the information theoretic metric learning (ITML) [9]. We consider a multi-class categorization problem. Suppose $(\boldsymbol{X}_1, \omega_1), \ldots, (\boldsymbol{X}_\ell, \omega_\ell)$ is a training dataset, where $\boldsymbol{X}_i$ is the covariance descriptor of the i-th example, and ω_i is its class label. From ℓ examples, K index pairs, $(i_1, j_1), \ldots, (i_K, j_K)$, are selected, and each pair is given the following constraint:

$$D_{\boldsymbol{\Phi}}(\boldsymbol{X}_{i_k}, \boldsymbol{X}_{j_k}; \boldsymbol{W}) \begin{cases} \leq b_{\mathrm{ub}} \xi_k, & \text{if } \omega_{i_k} = \omega_{j_k}, \\ \geq b_{\mathrm{lb}} \xi_k, & \text{if } \omega_{i_k} \neq \omega_{j_k}, \end{cases} \tag{2}$$

where two constants b_{ub} and b_{lb} denote the upper and lower bounds of the distances between two examples in the same and different classes, respectively, when $\xi_k = 1$. We define y_k and b_k for $k = 1, \ldots, K$ as $y_k := +1$ and $b_k := b_{\mathrm{ub}}$ for $\omega_{i_k} = \omega_{j_k}$, and as $y_k := -1$ and $b_k := b_{\mathrm{lb}}$ for $\omega_{i_k} \neq \omega_{j_k}$. Under constraint (2), we want to find $\boldsymbol{W}$ and ξ_k such that $\boldsymbol{W}$ is does not deviate significantly from the identity matrix and ξ_k is close to one. Based on this, we pose the following problem:

$$\begin{aligned} \min \quad & \mathrm{BD}_\varphi((\boldsymbol{W}, \boldsymbol{\xi}), (\boldsymbol{I}, \boldsymbol{1})), \\ \text{wrt} \quad & \boldsymbol{W} \in \mathbb{S}_{++}^n, \quad \boldsymbol{\xi} = [\xi_1, \ldots, \xi_K]^\top \in \mathbb{R}_{++}^K, \\ \text{subject to} \quad & \forall k \in \mathbb{N}_K, \quad y_k D_{\boldsymbol{\Phi}}(\boldsymbol{X}_{i_k}, \boldsymbol{X}_{j_k}; \boldsymbol{W}) \leq y_k b_k \xi_k, \end{aligned} \tag{3}$$

where $\mathrm{BD}_\varphi(\cdot, \cdot) : (\mathbb{S}_{++}^n \times \mathbb{R}_{++}^K) \times (\mathbb{S}_{++}^n \times \mathbb{R}_{++}^K) \to \mathbb{R}_+$ is the *Bregman divergence*. The divergence, $\mathrm{BD}_\varphi((\boldsymbol{W}, \boldsymbol{\xi}), (\boldsymbol{I}, \boldsymbol{1}))$, becomes zero if $(\boldsymbol{W}, \boldsymbol{\xi}) = (\boldsymbol{I}, \boldsymbol{1})$, and it increases if $(\boldsymbol{W}, \boldsymbol{\xi})$ deviates considerably from $(\boldsymbol{I}, \boldsymbol{1})$. The definition of the Bregman divergence contains a seed function, $\varphi : \mathbb{S}_{++}^n \times \mathbb{R}_{++}^K \to \mathbb{R}$, which is assumed to be continuously differentiable and strictly convex. For some φ, the Bregman divergence is defined as

$$\mathrm{BD}_\varphi(\boldsymbol{\Theta}, \boldsymbol{\Theta}_0) = \varphi(\boldsymbol{\Theta}) - \varphi(\boldsymbol{\Theta}_0) - \langle \nabla\varphi(\boldsymbol{\Theta}_0), \boldsymbol{\Theta} - \boldsymbol{\Theta}_0 \rangle,$$

for $\boldsymbol{\Theta}, \boldsymbol{\Theta}_0 \in \mathbb{S}_{++}^n \times \mathbb{R}_{++}^K$, where $\langle \cdot, \cdot \rangle$ denotes the inner product defined as,

$$\begin{aligned} &\forall (\boldsymbol{W}_1, \boldsymbol{\xi}_1), \forall (\boldsymbol{W}_2, \boldsymbol{\xi}_2) \in \mathbb{S}_{++}^n \times \mathbb{R}_{++}^K, \\ &\quad \langle (\boldsymbol{W}_1, \boldsymbol{\xi}_1), (\boldsymbol{W}_2, \boldsymbol{\xi}_2) \rangle = \langle \boldsymbol{W}_1, \boldsymbol{W}_2 \rangle + \langle \boldsymbol{\xi}_1, \boldsymbol{\xi}_2 \rangle. \end{aligned}$$

This implies that the magnitudes of the deviations of the solution, $(\boldsymbol{W}, \boldsymbol{\xi})$, from $(\boldsymbol{I}, \boldsymbol{1})$ depend on the definition of the

seed function. In this study, the seed function is assumed to be the sum of two terms, as follows:

$$\varphi(\boldsymbol{W}, \boldsymbol{\xi}) := \varphi_{\mathrm{r}}(\boldsymbol{W}) + \sum_{k=1}^{K} c_k \varphi_{\mathrm{l}}(\xi_k),$$

where c_k is a positive constant that trades off the importance of regularization versus that of losses. Larger c_k yields a harder margin. The first term in the definition of the seed function is defined as $\varphi_{\mathrm{r}} : \mathbb{S}^n_{++} \to \mathbb{R}$, where $\varphi_{\mathrm{r}}(\boldsymbol{W}) := -\mathrm{logdet}(\boldsymbol{W})$. For the second term containing $\varphi_{\mathrm{l}} : \mathbb{R}_{++} \to \mathbb{R}$, we considered the following three functions:

$$\varphi_{\mathrm{is}}(\xi_k) := -\log(\xi_k), \qquad \varphi_{\mathrm{l2}}(\xi_k) := \frac{1}{2}\xi_k^2,$$
$$\varphi_{\mathrm{e}}(\xi_k) := (\log \xi_k - 1)\xi_k.$$

The Bregman divergences generated from the three seed functions, φ_{is}, φ_{l2}, and φ_{e}, are referred to as the *Itakura–Saito Bregman divergence* (**ISBD**), *L2 Bregman divergence* (**L2BD**), and *Relative entropy Bregman divergence* (**REBD**), respectively, where the ISBD is equal to the objective function employed by Huang et al. [2].

3. Stochastic Variants of Dykstra Algorithm

We introduce the Dykstra algorithm [12], [13] to solve the optimization problem (3). The original Dykstra algorithm [13] was developed as a computational method that finds the Euclidean projection from a point onto the intersection of half-spaces. Additionally, the Dykstra algorithm can be applied for finding the projection onto the interaction of multiple convex sets. However, in this study, we focus on the interaction of half-spaces with the aim of utilizing the Dykstra algorithm for solving the proposed metric learning problem. Censor & Reich [12] extended the Dykstra algorithm for finding the *Bregman projection* from a point, $\boldsymbol{x}_0$, to an intersection, C, defined by

$$\operatorname*{argmin}_{\boldsymbol{x} \in C} \mathrm{BD}_{\varphi}(\boldsymbol{x}, \boldsymbol{x}_0). \tag{4}$$

In each iteration, the Dykstra algorithm selects a half-space in cyclic order to project the current solution onto the half-space (Fig. 1). The Dykstra algorithm ensures linear convergence to a minimum.

In existing literature related to stochastic gradient descent methods and variants [20]–[23] that minimize the regularized loss averaged over a set of examples, it has been shown empirically that example selection in stochastic order, instead of cyclic order, significantly accelerates convergence to the optimal solution.

Based on these facts, this study proposes the use of stochastic selection of half-spaces in the Dykstra algorithm, referred to as the *stochastic Dykstra algorithm*. The following three methods can be used to select half-spaces: **Cyclic**: A half-space is selected in cyclic order at each iteration. **Rand**: A half-space is selected arbitrarily at each iteration. **Perm**: Prior to selection, the orders of K half-spaces are permuted arbitrarily at the beginning of each epoch. We employ the "Rand" option, even though replacing this option with one of the remaining two options is straightforward.

If we define the k-th half-space, C_k, as $C_k := \{\boldsymbol{x} \mid \langle \boldsymbol{a}_k, \boldsymbol{x} \rangle \le b_k\}$, it can be shown, using the Lagrangian theory, that computing the Bregman projection from a point, $\boldsymbol{x}_0$, to its boundary, $\mathrm{bd}(C_k)$, is equivalent to solving the following saddle point problem:

$$\max_{\delta} \min_{\boldsymbol{x}} \mathrm{BD}_{\varphi}(\boldsymbol{x}, \boldsymbol{x}_0) + (\langle \boldsymbol{a}_k, \boldsymbol{x} \rangle - b_k)\delta \tag{5}$$

where $\delta \in \mathbb{R}$ is the Lagrangian multiplier.

Algorithm 1 Stochastic Dykstra Algorithm.

1: **begin**
2: $\forall k \in \mathbb{N}_K : \ \alpha_k := 0$;
3: **for** $t = 1, 2, \ldots$ **do**
4: Pick k randomly from $\{1, \ldots, K\}$;
5: Solve the following saddle point problem and let $\delta_{t-1/2}$ be the solution of δ:

$$\max_{\delta} \min_{\boldsymbol{x}} \mathrm{BD}_{\varphi}(\boldsymbol{x}, \boldsymbol{x}_{t-1}) + \delta_t(\langle \boldsymbol{a}_k, \boldsymbol{x} \rangle - b_k); \tag{6}$$

6: $\delta_t := \max(\delta_{t-1/2}, -\alpha_k)$; $\alpha_k := \alpha_k + \delta_t$;
7: $\boldsymbol{x}_t = \nabla\varphi^*(\nabla\varphi(\boldsymbol{x}_{t-1}) - \delta_t \boldsymbol{a}_k)$;
8: **end for**
9: **end.**

This enables us to rewrite the Dykstra algorithm with the Rand option for finding the Bregman projection from a point, $\boldsymbol{x}_0$, to the intersection of $C_1, \ldots, C_K$, as described in Algorithm 1, where φ^* is the convex conjugate [24] of the seed function, φ. When applying Algorithm 1 to metric learning problem (3), Step 5, which solves saddle point problem (6), is the most important step. In this study, a high-speed computational technique was devised for this step, whose details are presented in the next section.

4. Efficient Projection Technique

We first show that solving optimization problem (3) is equivalent to finding the Bregman projection from a point, $(\boldsymbol{I}, \boldsymbol{1}) \in \mathbb{S}^n_{++} \times \mathbb{R}^K_{++}$, onto the intersection of multiple half-spaces. Then, we propose a new high-speed technique for solving sub-problem (6).

Let $\boldsymbol{A}_k$ be a positive semi-definite matrix expressed as

$$\boldsymbol{A}_k := \Big(\boldsymbol{\Phi}(\boldsymbol{X}_{i_k}) - \boldsymbol{\Phi}(\boldsymbol{X}_{j_k})\Big)\Big(\boldsymbol{\Phi}(\boldsymbol{X}_{i_k}) - \boldsymbol{\Phi}(\boldsymbol{X}_{j_k})\Big)^{\top} \tag{7}$$

for $k \in \mathbb{N}_K$, to define a half-space

$$C_k := \Big\{(\boldsymbol{W}, \boldsymbol{\xi}) \in \mathbb{S}^n_{++} \times \mathbb{R}^K_{++} \,|\, y_k \langle \boldsymbol{A}_k, \boldsymbol{W} \rangle - y_k b_k \xi_k \le 0\Big\}. \tag{8}$$

Then, it can be shown that the intersection of K half-spaces, $\bigcap_{k=1}^K C_k$, is the feasible region of optimization problem (3). This implies that the Dykstra algorithm can be applied to solve problem (3). We assume that $\boldsymbol{A}_k$ is strictly positive

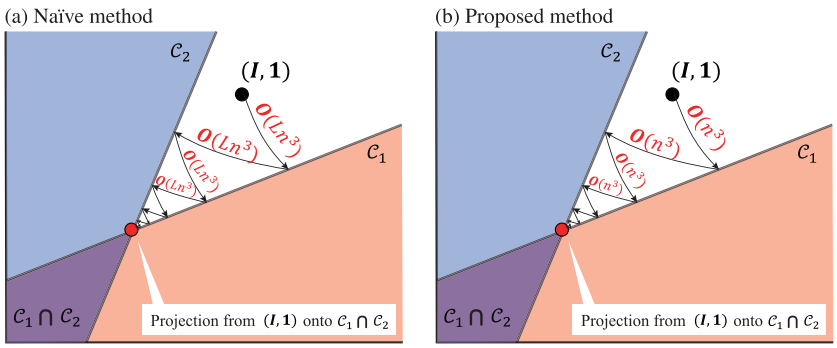


Fig. 1 Dykstra algorithm chooses a half-space in each iterate to project a current solution to the half-space. This study devised a new technique that performs the exact projection with $O(n^3)$ computation, where naïve method requires $O(Ln^3)$ computation, where L is defined in Sect. 4.

definite. This assumption can be satisfied by considering $\boldsymbol{A}_k \leftarrow \boldsymbol{A}_k + \varepsilon \boldsymbol{I}$, where ε is a small positive constant.

Then, we propose an efficient technique for Step 5 in Algorithm 1, which projects $(\boldsymbol{W}_{t-1}, \boldsymbol{\xi}_{t-1}) \in \mathbb{S}^n_{++} \times \mathbb{R}^K_{++}$ onto the k-th half-space, C_k, where $(\boldsymbol{W}_{t-1}, \boldsymbol{\xi}_{t-1}) \in \mathbb{S}^n_{++} \times \mathbb{R}^K_{++}$ is the model parameter after the $(t-1)$-th iteration. Performing the projection is equivalent to finding the saddle point of the function, $Q : \mathbb{S}^n_{++} \times \mathbb{R}^K_{++} \times \mathbb{R} \to \mathbb{R}$, defined as

$$Q(\boldsymbol{W}, \boldsymbol{\xi}, \delta) := \mathrm{BD}_\varphi((\boldsymbol{W}, \boldsymbol{\xi}), (\boldsymbol{W}_{t-1}, \boldsymbol{\xi}_{t-1})) + \delta y_k \left(\langle \boldsymbol{A}_k, \boldsymbol{W} \rangle - b_k \xi_k\right).$$

Lemma 4.1: Let $\xi_{k,t-1}$ be the k-th entry in vector $\boldsymbol{\xi}_{t-1}$. The value of the function, $J_t : \mathbb{R} \to \mathbb{R}$, defined by

$$J_t(\delta) := \left\langle \boldsymbol{A}_k, (\boldsymbol{W}_{t-1}^{-1} + \delta y_k \boldsymbol{A}_k)^{-1} \right\rangle - b_k \nabla\varphi_1^*(\nabla\varphi_1(\xi_{t-1}) + \delta y_k b_k / c_k), \tag{9}$$

is zero at the solution, δ, of the saddle point of Q. The solution, δ, must satisfy the strictly positive definiteness,

$$\boldsymbol{W}_{t-1}^{-1} + \delta y_k \boldsymbol{A}_k \succ \boldsymbol{O}, \tag{10}$$

and the feasibility of the slack variables,

$$\exists \xi_{k,t-1/2} \quad \text{s.t.} \quad \nabla\varphi_1(\xi_{k,t-1/2}) = \nabla\varphi_1(\xi_{k,t-1}) - \delta y_k b_k / c_k. \tag{11}$$

No closed-form solution was found for this projection problem. Hence, numerical methods, such as the Newton–Raphson method, are necessary for solving the nonlinear equation, $J_t(\delta) = 0$. Computing the value of $J_t(\delta)$ naïvely requires $O(n^3)$ computational cost because $J_t(\cdot)$ involves computation of the inverse of an $n \times n$ matrix. If we assume that the numerical method calculates the value of the scalar-valued function, $J_t(\cdot)$, L times, the naïve approach will require $O(Ln^3)$ computational time to find the solution of the nonlinear equation, $J_t(\delta) = 0$. Furthermore, the positive definiteness condition in (10) and the feasibility condition in (11) must be checked.

The proposed technique for finding the saddle point is given as follows: Let $d_1, \ldots, d_n$ be the eigenvalues of $\boldsymbol{A}_k^{-1/2} \boldsymbol{W}_{t-1}^{-1} \boldsymbol{A}_k^{-1/2}$ with $d_1 \geq \cdots \geq d_n$. Then, we obtain

$$J_t(\delta) = \sum_{i=1}^n \frac{1}{d_i + y_k \delta} - b_k \nabla\varphi_1^*\left(\nabla\varphi_1(\xi_{t-1}) + \frac{\delta y_k b_k}{c_k}\right), \tag{12}$$

which implies $J_t(\delta)$ can be assessed within $O(n)$ computational cost after $d_1, \ldots, d_n$ are obtained (the derivation of (12) is given in Sect. A.1). Considering this with the fact that it requires $O(n^3)$ time to obtain the n scalars, $d_1, \ldots, d_n$, the solution can be computed in $O(n^3 + Ln)$ time. We define $\delta_\mathrm{b} := c_k / (b_k \xi_{k,t-1})$. The interval of $y_k \delta$ satisfying (10) and (11) is given as $(-d_n, \delta_\mathrm{b})$ for ISBD, and $(-d_n, +\infty)$ for L2BD and REBD. The theoretical results are summarized in the following theorem:

Theorem 4.1: The saddle point of $Q(\boldsymbol{W}, \boldsymbol{\xi}, \delta)$ can be found within $O(n^3 + Ln)$ time, where L is the number of times that a numerical method calculates the value of $J_t(\cdot)$. Moreover, a unique solution exists.

As $L \in O(n^2)$ in a typical setting, we can say that each update can be performed in $O(n^3)$ computational time.

Convex Conjugate Functions: The proposed algorithm requires the derivative of the convex conjugate of the seed function, φ_l. The convex conjugates of the seed functions generating ISBD, L2BD, and REBD are given as

$$\varphi_\mathrm{is}^*(g) = \begin{cases} +\infty, & \text{for } g \geq 0, \\ -1 - \log(-g), & \text{for } g < 0, \end{cases} \tag{13}$$

$$\varphi_\mathrm{l2}^*(g) := \frac{1}{2} g^2, \qquad \varphi_\mathrm{e}^*(g) := \exp(g), \tag{14}$$

respectively. These derivatives are expressed as

$$\nabla\varphi_\mathrm{is}^*(g) = \frac{1}{g}, \quad \nabla\varphi_\mathrm{l2}^*(g) = g, \quad \nabla\varphi_\mathrm{e}^*(g) = \exp(g). \tag{15}$$

Stopping Criterion: We discuss how to determine if the solution is already optimal and when to terminate the algorithm. While running the algorithm, $(\boldsymbol{W}_t, \boldsymbol{\xi}_t)$ may violate a few constraints. The index set of the violated constraints is denoted by $\mathcal{I}_\mathrm{vio} := \{k \in \mathbb{N}_K \,|\, (\boldsymbol{W}_t, \boldsymbol{\xi}_t) \notin C_k\}$, and $\bar{\boldsymbol{\xi}}_t \in \mathbb{R}^K_{++}$ is defined such that the k-th entry is given by

$\bar{\xi}_{h,t} := \frac{1}{b_h}\langle \boldsymbol{W}_t, \boldsymbol{A}_h\rangle$ for $h \in \mathcal{I}_{\text{vio}}$ and $\bar{\xi}_{h,t} := \xi_{h,t}$ for $h \notin \mathcal{I}_{\text{vio}}$. Note that $(\boldsymbol{W}_t, \bar{\boldsymbol{\xi}}_t)$ is a feasible solution, and $\bar{\boldsymbol{\xi}}_t = \boldsymbol{\xi}_t$ when $(\boldsymbol{W}_t, \boldsymbol{\xi}_t)$ is feasible. The objective gap after iteration t is bounded as follows:

$$\begin{aligned}&\mathrm{BD}_\varphi((\boldsymbol{W}_t, \bar{\boldsymbol{\xi}}_t), (\boldsymbol{I}, \boldsymbol{1})) - \mathrm{BD}_\star \le \\ &\sum_{h\in\mathcal{I}_{\text{vio}}} c_h(\varphi_1(\bar{\xi}_{h,t}) - \varphi_1(\xi_{h,t}) - \nabla\varphi_1(1)(\bar{\xi}_{h,t} - \xi_{h,t})) \\ &- \sum_{h=1}^{K} \alpha_h y_h \left(\langle \boldsymbol{A}_h, \boldsymbol{W}_t\rangle - b_h\xi_{h,t}\right),\end{aligned} \tag{16}$$

where $\mathrm{BD}_\star$ denotes the minimum objective value. Then, this upper bound of the objective gap can be used as the stopping criterion of the Dykstra algorithm.

Based on the above discussion, the proposed metric learning algorithm for covariance descriptors is summarized as Algorithm 2.

Algorithm 2 Proposed Metric Learning Algorithm with Covariance Descriptors.

Input: $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_K, \boldsymbol{b}, \boldsymbol{c}, \varphi, \epsilon$.
Output: ϵ-approximate solution of $(\boldsymbol{W}, \boldsymbol{\xi})$.
1: **begin**
2: $\forall k \in \mathbb{N}_K :\ \alpha_k := 0$; $\boldsymbol{W}_0 = \boldsymbol{I}$; $\boldsymbol{\xi} = \boldsymbol{1}_K$;
3: **for** $t = 1, 2, \ldots$ **do**
4: Pick k randomly from $\{1, \ldots, K\}$;
5: Find the saddle point of the function $L(\boldsymbol{W}, \boldsymbol{\xi}, \delta)$ by solving the nonlinear equation $J_t(\delta) = 0$ using (12), where the solution must satisfy $y_k\delta \in (-d_n, \delta_{\mathrm{b}})$ for ISBD, and $y_k\delta \in (-d_n, +\infty)$ for L2BD and REBD, where $\delta_{\mathrm{b}} = c_k/(b_k\xi_{k,t-1})$. Let $\delta_{t-1/2}$ be the solution of δ:
6: $\delta_t := \max(\delta_{t-1/2}, -\alpha_k)$; $\alpha_k := \alpha_k + \delta_t$;
7: $\boldsymbol{W}_t := (\boldsymbol{W}_{t-1} + \delta_t y_k \boldsymbol{A}_k)^{-1}$; $\xi_k := \nabla\varphi^*(\nabla\varphi(\xi_k) + \delta_t y_k b_k/c_k)$;
8: **if** (RHS of (16)) $\le \epsilon$ **then** terminate the algorithm;
9: **end for**
10: **end**

5. Experiments

We conducted experiments to assess the convergence speed of the proposed optimization algorithms, the total computational time for learning, and the generalization performance for pattern recognition.

5.1 Convergence Behavior of Optimization Algorithms

We examined the proposed algorithms for assessing convergence speed. Artificial datasets were generated as follows: Fifty matrices, $\boldsymbol{F}_k \in \mathbb{R}^{n\times n}$ ($k = 1, \ldots, K$ where $K = 50$), were generated, in which each entry was obtained from a uniform distribution in the interval $[-0.5, 0.5]$. Then, we set $\boldsymbol{A}_k := \boldsymbol{F}_k\boldsymbol{F}_k^\top$. The values of the variables, y_k, were arbitrarily selected from $\{\pm 1\}$ with the same probabilities. We set $\boldsymbol{b} = \boldsymbol{1}$ and $\boldsymbol{c} = \boldsymbol{1}/(\lambda K)$.

Figure 2 shows the convergence behavior of the cyclic Dykstra algorithm and the two stochastic Dykstra algorithms with $\lambda = 10^{-4}$ and $n = 100$. One epoch corresponds to projecting onto a half-space K times. Typically, ISBD converges faster than the other Bregman divergences. We report the result for ISBD. For most values of λ and n, the two stochastic Dykstra algorithms converge faster than the cyclic algorithm. Particularly, when regularization is weak (e.g., $\lambda = 10^{-4}$), the cyclic algorithm is considerably slow and cannot be put in practice. However, the stochastic version attains an accurate solution, as shown in Fig. 2. As the convergence performances of Rand and Perm do not differ significantly, only the experimental results obtained using Perm are reported.

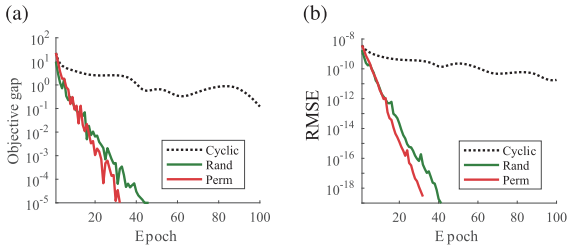


Fig. 2 Convergence behavior of Dykstra algorithms. The stochastic order reduced the objective gap $\mathrm{BD}_t - \mathrm{BD}_\star$ and the RMSE $\|\boldsymbol{W}_t - \boldsymbol{W}_\star\|_{\mathrm{F}}$ much faster, where BD_t is the objective value at time t and $\boldsymbol{W}_\star$ is the optimal value of $\boldsymbol{W}$.

5.2 Total Computational Time for Learning

Table 1 shows the computational times for metric learning of $\boldsymbol{W}$ and compares the proposed projection technique (Sect. 4) with the naïve method, where the proposed technique requires $O(n^3 + Ln)$ time for a projection, whereas the naïve method requires $O(Ln^3)$ time. The experimental settings in Table 1 are similar to those in Fig. 2. For obtaining the results shown in Table 3, we tested the data with $n = 5, 10, 50, 100, 500$ and $\epsilon = 10^{-3}$. The runtimes were measured on a Linux machine with an Intel(R) Core(TM) i7 (3.6GHz) CPU and 32 GB RAM. We repeated the experiments five times, and the average computational times are shown in Table 1. The proposed technique was faster than the naïve method in all cases, and significant improvements were achieved for larger values of n.

5.3 Generalization Performance for Pattern Recognition

We examined the generalization performance for the following two image recognition tasks: texture recognition and generic visual categorization.

For the texture classification task, we used the Brodatz texture dataset containing 112 different texture images. Each image had a size of 640×640 and was gray-scaled. Each image was divided into four sub-images of equal sizes. Two of the four sub-images were arbitrarily selected for testing, and the remaining images were used for training. For each training and testing image, the covariance descriptors of 20 arbitrarily selected images were extracted from 128×128 patches. Then, $4480(112 \times 2 \times 20)$ covariance descriptors were obtained for training and testing. For

Table 1 Computational times.

n	5	10	50	100	500
Proposed (sec)	**20.93**	**15.55**	**13.24**	**20.04**	**234.64**
Naïve (sec)	21.95	17.84	51.48	184.69	4166.01
# of Epochs	18.5	12.3	8.5	10.3	9.0

Table 2 Average accuracies (%) of the Proposed Method when using different losses and different mapping functions for pattern recognition.

(a) Brodatz

	Eye	REBD	L2BD	ISBD
Id	73.08	**79.55**	79.29	79.33
Log	79.29	**80.27**	80.18	79.96
Sqrt	76.88	80.94	80.89	**81.12**
Chol	78.71	82.77	**82.90**	82.72

(b) ETH-80

	Eye	REBD	L2BD	ISBD
Id	66.98	76.93	**77.99**	76.79
Log	94.40	96.12	**96.35**	96.12
Sqrt	88.26	88.96	**89.32**	88.74
Chol	90.88	91.64	**91.85**	90.83

Table 3 Mean recognition accuracies (%) with standard deviations. The underlined figures indicate that the performance is not significantly different from the best accuracy at the significance level 1% based on one-sample t-test.

	Euclid	Ours	V&J [3]	Huang et al [2]
Brodatz	79.29 ± 1.04	**82.90** ± 1.18	65.63 ± 2.18	79.33 ± 1.59
ETH-80	94.40 ± 0.55	**96.35** ± 0.49	93.40 ± 1.02	96.09 ± 0.44
EEG	55.04 ± 0.14	**56.76** ± 0.15	53.48 ± 0.16	55.57 ± 0.14

evaluating generalized performance, the k-nearest neighbor classifier was used, where the number of nearest neighbors was set as three. We set $K = 100 \times n_c$, where n_c is the number of classes. Following [9], we set $b_{\mathrm{ub}} = 0.05$, and $b_{\mathrm{lb}} = 0.95$. The reguralization parameter, λ, was chosen by cross-validation within the training dataset.

For the generic visual categorization task, the ETH-80 dataset containing $n_c = 8$ classes was used. Each class had 10 objects, each of which included 41 colored images. For every object, 20 images were arbitrarily selected for training, and the remaining images were used for testing.

One covariance matrix was obtained from each image. We used the following four types of $\boldsymbol{\Phi}$: **Id**: $\boldsymbol{\Phi}(\boldsymbol{X}) = \boldsymbol{X}$, **Log**: $\boldsymbol{\Phi}(\boldsymbol{X}) = \mathbf{logm}(\boldsymbol{X})$, **Sqrt**: $\boldsymbol{\Phi}(\boldsymbol{X}) = \boldsymbol{X}^{1/2}$, and **Chol**: $\boldsymbol{\Phi}(\boldsymbol{X}) = \mathbf{chol}(\boldsymbol{X})$. Parameter $\boldsymbol{W}$ is determined using the metric learning algorithms with ISBD, L2BD, and REBD, and compared with $\boldsymbol{W} = \boldsymbol{I}$, which we denote as **Eye**. Note that $D_{\boldsymbol{\Phi}}(\cdot, \cdot; \boldsymbol{I}) = D_{\boldsymbol{\Phi}}(\cdot, \cdot)$. Table 2 summarizes the average accuracies over ten repeated experiments. For each type of $\boldsymbol{\Phi}$, supervised metric learning improved the generalization performance for texture classification and for generic visual categorization. For texture classification, the Cholesky decomposition-based mapping, $\mathbf{chol}(\cdot)$, exhibited the highest accuracy, while the matrix logarithm-based mapping, $\mathbf{logm}(\cdot)$, exhibited the highest accuracy for generic image categorization.

We compared the proposed method with two existing metric learning methods proposed by Huang et al. [2] and Vemulapalli and Jacobs [3]. We used Huang et al.'s implementation available from their web site (http://www.vision.ee.ethz.ch/~zzhiwu/codes/LEML-v1.0.zip). The update rule that they used does not ensure convergence of the algorithm, and their study does not propose a stopping criterion. We implemented Vemulapalli and Jacobs's method in MATLAB. We performed principal component analysis after vectorization so that the degree of freedom in their study coincides with that in our and Huang et al.'s works. As shown in the first and second rows of Table 3, the proposed metric learning algorithm is more accurate than existing distances (1), and its performance surpasses that of the two existing metric learning methods.

In addition, we examined the proposed methods for EEG signal classification. We used the BCI Competition IV dataset IIa [25]. This dataset contains EEG signals recorded from nine subjects by attaching twenty-two electrodes to the scalp, while they perform four different motor imageries, including left hand, right hand, foot, and tongue, i.e., there are four classes in this task. We arbitrarily selected 30% of the data for testing and the remaining for training. This was repeated ten times to obtain average performances. We considered signals in a range of 0.5–2.5 seconds after the onset of each cue. We used parameters $b_{\mathrm{ub}} = \max(\mathrm{mean} - \mathrm{std}, 0.01)$ and $b_{\mathrm{lb}} = \mathrm{mean}+\mathrm{std}$, where mean and std represent the average and standard deviation, respectively, of the distances among the training data. We set other parameters to the same values as those used in the image classification tasks. Linear discriminant analysis was used as a classifier. The result is shown in the last row of Table 3. The accuracies represent the highest values among four mapping functions. **Sqrt** was the most suitable $\boldsymbol{\Phi}$ for Euclid, and a combination of **Log** and ISBD was the most suitable for the proposed methods. The performance of the proposed methods is slightly better than that of existing methods.

6. Conclusions

In this study, we have devised several objective functions for metric learning on a positive semidefinite cone, all of which can be minimized using the Dykstra algorithm.

We have proposed a new technique that performs each update efficiently. We have empirically demonstrated that the stochastic versions of the Dykstra algorithm are considerably faster than the original algorithm, and the generalization performance for pattern recognition improves significantly. A strong advantage of the proposed algorithm is the existence of a stopping criterion that ensures sub-optimality, even though a gradient method is frequently employed for non-linear optimization. Future work involves numerical comparison between our approach and the gradient approach.

References

[1] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M.T. Harandi, "Kernel methods on the Riemannian manifold of symmetric positive definite matrices," CVPR, pp.73–80, IEEE, 2013.
[2] Z. Huang, R. Wang, S. Shan, X. Li, and X. Chen, "Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification," ICML, pp.720–729, 2015.
[3] R. Vemulapalli and D.W. Jacobs, "Riemannian metric learning for symmetric positive definite matrices," arXiv:1501.02393, 2015.
[4] F. Yger and M. Sugiyama, "Supervised logeuclidean metric learning for symmetric positive definite matrices," arXiv:1502.03505, 2015.
[5] X. Pennec, P. Fillard, and N. Ayache, "A Riemannian framework for tensor computing," International Journal of Computer Vision, vol.66, no.1, pp.41–66, 2006.
[6] S. Sra, "A new metric on the manifold of kernel matrices with application to matrix geometric means," Advances in Neural Information Processing Systems, pp.144–152, 2012.
[7] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Log-Euclidean metrics for fast and simple calculus on diffusion tensors," Magnetic resonance in medicine, vol.56, no.2, pp.411–421, 2006.
[8] I.L. Dryden, A. Koloydenko, and D. Zhou, "Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging," The Annals of Applied Statistics, vol.3, no.3, pp.1102–1123, 2009.
[9] J.V. Davis, B. Kulis, P. Jain, S. Sra, and I.S. Dhillon, "Information-theoretic metric learning," ICML, pp.209–216, ACM, 2007.
[10] T. Kato and N. Nagano, "Metric learning for enzyme active-site search," Bioinformatics, vol.26, no.21, pp.2698–2704, Nov. 2010.
[11] R. Relator, N. Nagano, and T. Kato, "Using bregmann divergence regularized machine for comparison of molecular local structures," IEICE Transactions on Information & Systems, vol.E99-D, no.1, pp.275–278, Jan. 2016.
[12] Y. Censor and S. Reich, "The Dykstra algorithm with Bregman projections," Comm. Appl. Anal., vol.2, pp.407–419, 1998.
[13] R.L. Dykstra, "An algorithm for restricted least squares regression," Journal of the American Statistical Association, vol.78, no.384, pp.837–842, Dec. 1983.
[14] R. Sivalingam, V. Morellas, D. Boley, and N. Papanikolopoulos, "Metric learning for semi-supervised clustering of region covariance descriptors," Third ACM/IEEE International Conference on Distributed Smart Cameras, 2009 (ICDSC 2009), pp.1–8, 2009.
[15] D. Tosato, M. Farenzena, M. Spera, V. Murino, and M. Cristani, "Multi-class classification on Riemannian manifolds for video surveillance," Proceedings of the 11th European conference on Computer vision: Part II, Berlin, Heidelberg, vol.6312, pp.378–391, Springer-Verlag, 2010.
[16] I. Horev, F. Yger, and M. Sugiyama, "Geometry-aware principal component analysis for symmetric positive definite matrices," Proceedings of the Fourth Asian Conference on Machine Learning (ACML2015), JMLR Workshop and Conference Proceedings, vol.45, Hong Kong, China, pp.1–16, Nov. 2015.
[17] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J.S. Kandola, "On kernel-target alignment," NIPS, ed. T.G. Dietterich, S. Becker, and Z. Ghahramani, pp.367–373, MIT Press, 2001.
[18] T. Matsuzawa, R. Relator, J. Sese, and T. Kato, "Stochastic dykstra algorithms for metric learning with positive definite covariance descriptors," The 14th European Conference on Computer Vision (ECCV2016), vol.9910, pp.786–799, 2016.
[19] T. Matsuzawa, R. Relator, J. Sese, and T. Kato, "Stochastic dykstra algorithms for metric learning on positive semi-definite cone," Tech. Rep., arXiv:1601.01422, 2016.
[20] L. Bottou, "Large-scale machine learning with stochastic gradient descent," Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010), ed. Y. Lechevallier and G. Saporta, Paris, France, pp.177–187, Springer, Aug. 2010.
[21] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," Advances in Neural Information Processing Systems 26: Proceedings of a meeting held Dec. 2013, pp.315–323, Lake Tahoe, Nevada, United States, 2013.
[22] N.L. Roux, M. Schmidt, and F.R. Bach, "A stochastic gradient method with an exponential convergence _rate for finite training sets," in Advances in Neural Information Processing Systems 25, ed. F. Pereira, C. Burges, L. Bottou, and K. Weinberger, pp.2663–2671, Curran Associates, 2012.
[23] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: primal estimated sub-gradient solver for SVM," Math. Program., vol.127, no.1, pp.3–30, 2011.
[24] S. Boyd and L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[25] M. Tangermann, K.-R. Müller, A. Aertsen, N. Birbaumer, C. Braun, C. Brunner, R. Leeb, C. Mehring, K.J. Miller, G.R. Müller-Putz, G. Nolte, G. Pfurtscheller, H. Preissl, G. Schalk, A. Schlogl, C. Vidaurre, S. Waldert, and B. Blankertz, "Review of the bci competition iv," Front Neurosci, vol.6, p.55, 2012.

Appendix:

A.1 Derivation of Equation (12)

$d_1, \ldots, d_n$ are the eigenvalues of $\boldsymbol{A}_k^{-1/2}\boldsymbol{W}_{t-1}^{-1}\boldsymbol{A}_k^{-1/2}$. Let $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ have the corresponding n eigenvectors in its columns. We define a diagonal matrix, $\boldsymbol{D} := \mathrm{diag}(\{d_1, \ldots, d_n\})$. Then, we have $\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^\top = \boldsymbol{A}^{-1/2}\boldsymbol{W}_{t-1}^{-1}\boldsymbol{A}^{-1/2}$. The matrix, $\boldsymbol{W}$, at the saddle point can be expressed as

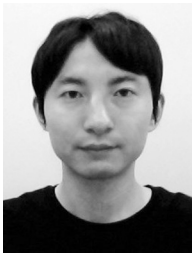
$$\begin{aligned}
\boldsymbol{W} &= (\boldsymbol{W}_{t-1} + \delta y_k \boldsymbol{A}_k)^{-1} \\
&= \boldsymbol{A}_k^{-1/2}\left(\boldsymbol{A}_k^{-1/2}(\boldsymbol{W}_{t-1} + \delta y_k \boldsymbol{A}_k)\boldsymbol{A}_k^{-1/2}\right)^{-1}\boldsymbol{A}_k^{-1/2} \\
&= \boldsymbol{A}_k^{-1/2}\left(\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^\top + \delta y_k \boldsymbol{I}_n\right)^{-1}\boldsymbol{A}_k^{-1/2} \\
&= \boldsymbol{A}_k^{-1/2}\boldsymbol{U}(\boldsymbol{D} + \delta y_k \boldsymbol{I}_n)^{-1}\boldsymbol{U}^\top\boldsymbol{A}_k^{-1/2}.
\end{aligned}$$

Substituting this into the first term of (9), we obtain

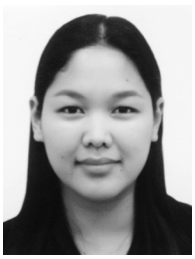
$$\begin{aligned}
&\left\langle \boldsymbol{A}_k, (\boldsymbol{W}_{t-1} + \delta y_k \boldsymbol{A}_k)^{-1}\right\rangle \\
&= \left\langle \boldsymbol{A}_k, \boldsymbol{A}_k^{-1/2}\boldsymbol{U}(\boldsymbol{D} + \delta y_k \boldsymbol{I}_n)^{-1}\boldsymbol{U}^\top\boldsymbol{A}_k^{-1/2}\right\rangle \\
&= \left\langle \boldsymbol{I}, (\boldsymbol{D} + \delta y_k \boldsymbol{I}_n)^{-1}\right\rangle = \sum_{i=1}^{n}\frac{1}{d_i + y_k\delta}.
\end{aligned}$$

The second term of (9) is the same as that of (12). Thus, the equality of (12) is derived.

Tomoki Matsuzawa received the B.E. from Gunma University in 2015, and is currently pursuing his M.E. degree at the Graduate School of Science and Engineering, Gunma University. His research interests include machine learning and computer vision.



Eisuke Ito received the B.E. and M.E. from Gunma University in 2012 and 2014, respectively, and is currently pursuing a PhD degree at the Graduate School of Science and Engineering, Gunma University. His research interests include neuroscience, machine learning and computer vision. He is a student member of JSAI.



Raissa Relator received the B.S. and M.S. degrees in Mathematics from the University of the Philippines Diliman in 2005 and 2008, respectively, and Ph.D. degree in Computer Science from Gunma University in 2015. She is currently a postdoctoral researcher at the National Institute of Advanced Industrial Science and Technology. Her research interests include machine learning, bioinformatics, statistical genetics and genetic epidemiology.

Jun Sese received the B.S., M.S. and Ph.D. from University of Tokyo at 1999, 2001 and 2005, respectively. He worked at the University of Tokyo as a research associate from 2003 to 2006, at Ochanomizu University as an associate professor from 2006 to 2011, and at Tokyo Institute technology as an associate professor from 2011 to 2014. He is working at AIST from 2014. His interest is to develop new machine learning methods and their application to life science.

Tsuyoshi Kato received his B.E., M.E., and Ph.D. degrees from Tohoku University, Sendai, Japan, in 1998, 2000, and 2003, respectively. From 2003 to 2005, he was with the National Institute of Advanced Industrial Science Technology (AIST) as a postdoctoral fellow in the Computational Biology Research Center (CBRC) in Tokyo. From 2005 to 2008, he was an assistant professor at the Graduate School of Frontier Sciences, University of Tokyo. From 2008 to 2010, he was an associate professor at the Center for Informational Biology, Ochanomizu University. He is now an associate professor at the Graduate School of Engineering, Gunma University. His current scientific interests include pattern recognition, computer vision, water engineering and bioinformatics. He is a member of IEICEJ.